

(19) World Intellectual Property Organization  
International Bureau(43) International Publication Date  
28 December 2000 (28.12.2000)

PCT

(10) International Publication Number  
WO 00/79365 A2(51) International Patent Classification<sup>7</sup>:

G06F

(74) Agent: NOWAK, Keith, D.; Lieberman & Nowak, LLP,  
350 Fifth Avenue, New York, NY 10118 (US).

(21) International Application Number:

PCT/US00/40279

(22) International Filing Date:

21 June 2000 (21.06.2000)

(81) Designated States (national): AG, AL, AM, AT, AU, AZ,  
BA, BB, BG, BR, BY, BZ, CA, CH, CN, CU, CZ, DE, DK,  
DZ, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP,  
KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD,  
MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU,  
SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ,  
VN, YU, ZW.

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

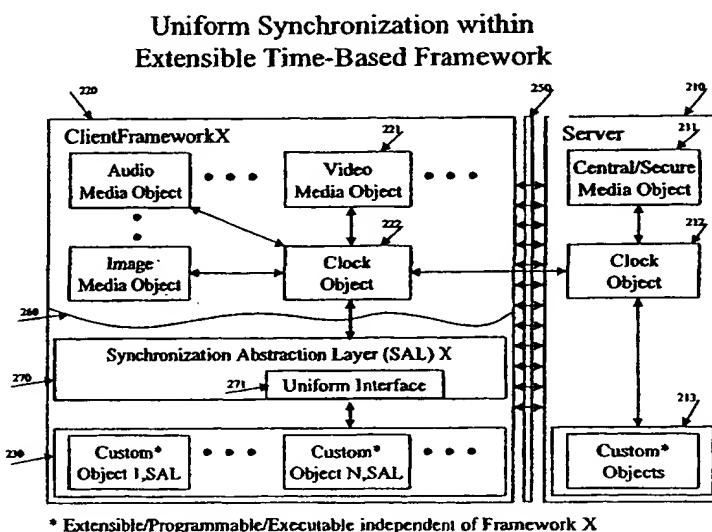
09/338,188 22 June 1999 (22.06.1999) US

(71) Applicant: SOFTCOM, INC. [US/US]; 110 Wall Street,  
2nd Floor, New York, NY 10005 (US).(72) Inventors: WASON, Andrew; 6 Victorian Woods Drive,  
Atlantic Highland, NJ 07716 (US). MILLS, Michael; 243  
Hamilton Drive, Red Bank, NJ 07701 (US). O'BRIEN,  
Chris; Suite 33, 98 Thompson Street, New York, NY  
10012 (US). WALLACE, Bruce, A.; 15 Ferncliff Terrace,  
Short Hills, NJ 07078 (US).(84) Designated States (regional): ARIPO patent (GH, GM,  
KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian  
patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European  
patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE,  
IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG,  
CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).**Published:**

- Without international search report and to be republished upon receipt of that report.

*[Continued on next page]*

(54) Title: CROSS-PLATFORM FRAMEWORK-INDEPENDENT SYNCHRONIZATION ABSTRACTION LAYER



(57) Abstract: This invention is a synchronization abstraction layer (SAL) providing a uniform interface to frameworks operating on sequenced flow data. It allows content developers to design and build interactive content that will operate interchangeably in different multimedia frameworks (e.g., Apple Computer, Inc.'s QuickTime™, Microsoft Corporation's NetShow™, RealNetworks, Inc.'s RealPlayer™, Sun Microsystems, Inc.'s Java™ Media Framework) and on different hardware platforms (e.g., desktop PC, Macintosh™, Television set-top boxes such as those from General Instrument Corporation and Scientific Atlanta Inc., Inc., Internet appliances such as AOL™-TV, and other appliances, e.g., a kitchen Internet radio). The uniform interface is independent of the particular framework and the platform on which the SAL is implemented, so that a single instance of content, whether created in Java™, JavaScript, VBscript, HTML, XML,

or some other language, can run appropriately on different hardware, e.g., on a Television set-top and on a desktop PC. In one realization, the synchronization abstraction layer provides a Java™ VIRTUAL MACHINE (JVM) interface for running Java™ plug-ins for streaming media applications such as Real Networks, Inc.'s RealPlayer™, Microsoft Corporation's Windows Media Technologies (NetShow™), Apple Computer, Inc.'s QuickTime™, Sun Microsystems, Inc.'s Java™ Media Framework. The JVM interface allows third-party developers to design platform- and framework-independent plug-ins for streaming media applications. This invention allows content providers to use plug-ins or compatible software objects (such as Java™ applets) to build, for example, interactive streaming media content that is fully interactive but independent of the particular underlying hardware and software technologies, such as RealNetworks™ G2, Microsoft Corporation's NetShow™, a desktop PC, or a television.

WO 00/79365 A2



*For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.*

**CROSS-PLATFORM FRAMEWORK-INDEPENDENT  
SYNCHRONIZATION ABSTRACTION LAYER**

5      **APPENDIX AND COPYRIGHT NOTICE**

This document includes a partial source code listing of a preferred embodiment of the applicant's invention. The code, listed in Appendix A and in the drawings, forms part of the disclosure of the specification. It is copyrighted. The copyright owner has no objection to facsimile reproduction by anyone of the patent document, including the copyrighted Appendix A and the drawings as they appear in the Patent and Trademark Office file or records, but otherwise reserves all rights.

15      **TECHNICAL FIELD**

This invention relates to the field of software plug-ins for multimedia file players and for other applications supporting ordered data flow files. More precisely, this invention defines a new field of software that allows plug-ins and content to be insulated from differences in underlying platforms and frameworks.

25      **BACKGROUND OF THE INVENTION**

Multimedia player frameworks have become widespread. Later versions of the most popular Internet browsers - Microsoft's Internet Explorer and Netscape's Communicator - include at least one player in the basic package. These are frameworks such as

RealNetworks, Inc.'s RealPlayer™ G2 family (collectively, the "RealPlayer™"); Microsoft Corporation's Windows Media Technologies (NetShow™); Macromedia, Inc.'s Director™; Apple Computer, Inc.'s QuickTime™; and Sun Microsystems, Inc.'s Java™ Media Framework (JMF).  
5 Most of these frameworks are extensible by means of plug-ins discussed below; some, notably JMF, are extensible by means of applications built on top of the frameworks. By being extensible we mean that a particular framework supports a set of interfaces exposed for interaction with additional software modules or  
10 components.

A framework is a set of interfaces, e.g., API's and procedures, and a set of capabilities exposed for the purposes of extensibility. It need not operate on multimedia-type files, i.e., files ordered sequentially and supporting the concept of a timeline; frameworks generally need not operate on time- or otherwise-ordered data files. In the rest of this document, however, we will discuss predominantly frameworks operating on ordered data flow files. We will refer to such frameworks interchangeably as "frameworks," "extensible frameworks," "extensible data flow-based frameworks," or by similar expressions; we do not imply any  
15 difference in the particular designation used, unless otherwise indicated.  
20

Frameworks are extended by means of extensions, for example plug-ins. Plug-ins, also referred to as extension modules, are

modules or components dynamically loaded at run-time. Extensible architecture and plug-ins have been used in commercial products, the RealPlayer™ and Windows Media Technologies being two instances. They also have been described in various publications. See United States Patents No. 5,838,906 and 5,870,559. (The specifications of these patents are incorporated by reference as if fully set forth herein.) A number of companies market plug-ins specifically for extending multimedia players.

10 **OBJECTS OF THE INVENTION**

One difficulty faced by plug-in developers is that a plug-in must be ported for each platform, i.e., for each hardware-operating system combination. Another difficulty lies in adapting a plug-in to various frameworks. Third difficulty, closely related to the first two, is that platform porting and framework adaptation require developers to have working knowledge of the various platforms and frameworks. One object of this invention is to provide a single interface that allows plug-in developers to build a single, platform independent version of a plug-in. Another object of this invention is to provide a uniform interface to the various frameworks so that a single plug-in can extend functionality of multiple frameworks. The third object of this invention is to facilitate development of content that can operate similarly, i.e., consistently, with different multimedia players

WO 00/79365

(e.g., RealNetworks, Inc.'s RealPlayer™ and Microsoft Corporation's Windows Media Technologies (NetShow™)), and platforms (TV, PC, set-top boxes). The fourth object of this invention invention is to accelerate the development of content and multimedia plug-ins by promoting reuse of existing software objects (e.g., objects written 5 in Java™ and JavaScript).

SUMMARY OF THE INVENTION

This invention is an abstraction layer providing a uniform interface between a framework and one or more plug-ins. In the 10 preferred embodiment, the invention is a Synchronization Abstraction Layer (SAL) abstracting time-based frameworks into a common synchronization interface. The SAL synchronizes itself and other plug-ins to a time-line of the underlying framework - and it does that independently of the underlying framework. In other 15 words, the plug-ins interact with the underlying framework through the SAL, rather than directly. Typically, the SAL is implemented on top of the synchronization of the Application Programming Interfaces (API's) provided by the underlying frameworks. It has 20 at least one point of coupling with the underlying framework: a method for providing the SAL with the current time.

The SAL includes a uniform cross-platform interface for the plug-ins. Preferably, the cross-platform interface has a functional core independent of the specific framework underlying

the SAL.

Some of the features of this invention are listed below:

1. Secure e-commerce, full interactive experiences including text and voice chat, games, graphics, animations, and advertising can be integrated and synchronized with streaming multimedia such as video and audio;
2. It can be used to build a content framework that insulates content and plug-in developers from details and differences in hardware platforms, so that the same content or plug-in can run on desktop platforms (PC, Macintosh, Solaris, Linux), Televisions (GI, SA), or other kinds of devices (AOL-TV, screen phones);
3. It can be used to build a content framework that insulates content and plug-in developers from specifics of, or differences in, software platforms, so that the same content or plug-in can run on Microsoft NetShow™, RealNetworks RealPlayer™, Apple Quicktime™, Sun Java™ Media Framework or any other media system;
4. It can run without a network or on different types of networks, such as wireless, Intranets, or the Internet;
5. It can be used to synchronize arbitrary data types,

WO 00/79365

including text, chat, graphics, video, audio, and active content like Java™, JavaScript, or Flash;

6. The framework-independent layer can be implemented using  
5 different languages, including Java™, HTML, XML, JavaScript,  
VBScript, or other languages;

7. When used for video synchronization, it can be used to  
synchronize arbitrary datatypes with different spatial objects in  
10 video, and with different temporal objects in video.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a diagram of basic processes in a traditional extensible time-based framework of prior art.

15 Figure 2 is a diagram of basic processes in an extensible time-based framework incorporating the Synchronization Abstraction Layer in accordance with this invention.

Figure 3 is a high level diagram of a preferred embodiment implementation specific to a table of contents plug-in.

20 Figure 4 lists a sample toc.rj, an XML descriptor file for a table of contents plug-in of the previous Figure.

Figure 5 lists a sample toc.xml, an XML descriptor file with detailed hierarchical description of the table of contents plug-in of Figure 3.

Figure 6 lists a sample toc.smi, an XML file defining a set of synchronized plug-ins.

DETAILED DESCRIPTION OF THE INVENTION

5

Figure 1 depicts conventional operation of an extensible multimedia player framework connected to a media server. Server 110 includes Media Object 111, Clock Object 112, and Custom Objects 113. Framework 120 and Extension Module (plug-in) 130 coexist on the client side. As is usual for a media player, Framework 120 contains Video Media Object 121, Clock Object 122, and Audio Media Object 123. Extension Module 130 has Custom Objects 131 and 132. Server 110 connects to Framework 120 and Extension Module 130 through Network 150. Line 160 indicates the extensibility interface between Framework 120 and Extension Module 130. Note that this high level diagram of prior art omits many other components and processes, including an operating system and rendering means.

10

Figure 2 demonstrates an arrangement functionally similar to the arrangement of Figure 1, but using the present invention. Server 210, including Media Object 211, Clock Object 212, and Custom Objects 213, connects to Framework 220, Extension Module (plug-in) 230, and Synchronization Abstraction Layer (SAL) 270 through Network 250. Line 260 symbolizes the interface extending Framework 220 through SAL 270. Block 271 denotes Uniform

Interface between SAL 270 and various extension modules, including Extension Module 230.

As before, Figure 2 omits many other components and processes, and is rather generalized. But a number of structural changes can be envisioned even in this, high level representation. For instance, media files can be stored locally, obviating the need for transmission over the network. Further, custom objects need not reside on the same server as the media files; they may be located on a separate server or, in some cases, locally. Several extension modules can be simultaneously connected to the same client framework, either through the SAL or directly. (The SAL itself is, of course, an extension module.)

The discussion above focused on media players rendering multimedia files supporting the concept of time-line, such as video and audio clips, because these applications are commercially important. The invention, however, is not limited to rendering multimedia or time-line ordered files. Time-line sequencing is only one example of data flow order within files, including media files. Many other types of ordering are possible. As one example, consider a video taken from a window of a car going down a highway. The images can be time- or distance-sequenced.

Files of interest in the present context, which we designate as ordered data flow files, may also be ordered in multiple

dimensions. Imagine a video game called "Tank Commander," with the player controlling a virtual tank moving cross country. Because the tank can move in two dimensions, the top view of the tank and its immediate surroundings depends on two coordinates.

5 The video data must therefore be ordered in two dimensions.

Similarly, video of a one-directional view from an object moving in three dimensions requires three dimensional ordering.

Clearly, the concept of ordered files goes beyond the four dimensions of space-time.

10 We also need not limit this concept to files ordered according to variables perceived as continuous, e.g., space and time. The concept is equally applicable to ordering according to discrete variables, such as event occurrences or quantized space positions. As an example, consider a server storing a library of books and sending the content to the client for sequential viewing page by page. The concept of ordered data flow files is therefore a general one.

15 Finally, in its general form the invention need not synchronize data flow, but may merely provide a uniform cross-platform interface for plug-ins.

20 Although the invention can be implemented in various languages, it is preferable to use a platform-independent source code. Therefore, consistent with the spirit of the invention, its preferred embodiment is implemented in Java™ - an

WO 00/79365

architecture-neutral, object-oriented, multithreaded language intended for use in distributed environments. (Partial source code of the preferred embodiment is listed in Appendix A and constitutes a part of the disclosure of this specification.) The Java™ code is executed by a Java™ Virtual Machine (JVM), which is a byte-code interpreter. The JVM is stored in one location and instantiated separately for each extension module, eliminating the need to store duplicate copies of the JVM for other modules that are also implemented in Java™.

10 Alternatively, the invention can instantiate a JVM from a copy in the framework, in the operating system, or even in another application.

The preferred embodiment creates a uniform interface to the RealPlayer™ of Real Networks. It can be easily adapted to extend other multimedia players in addition to the RealPlayer™, especially if a common set of functional features exists in the various players. This should be the case for most multimedia players. Because the Java™ code of a specific extension can be mostly independent of a particular framework being extended, large sections of the SAL's code can be shared among different framework adaptations of the same plug-in.

20 Figure 3 illustrates a high level diagram of a more specific example of the preferred embodiment. Here, the SAL is used to plug-in a table of contents ("TOC") extension module into the

RealPlayer™. The TOC plug-in displays a hierarchical view of the table of contents of a video presentation described in the table of contents. In the hierarchy, entries in the table are highlighted during the video presentation of their associated video portions. A user can randomly access various portions of the video by clicking on the corresponding entries in the TOC.

5 In this configuration RealPlayer™ 301 has three extension modules: SAL (RealJava) 310, RealVideo 302, and RealText 303. (Hereinafter we denote by "rj" or "RealJava" all Java™ objects compatible with the G2 architecture.) For clarity, SAL 10 310 is expanded within the dotted oval into its conceptual components - File Format Filter 311 and Renderer 312. This conceptual separation results from the G2 architecture constraints of the RealPlayer™; in a SAL adaptation compatible 15 with another media player, such conceptual separation may be inappropriate.

When a user invokes the RealPlayer™ application and the TOC extension module, RealPlayer™ 301 accesses Web Server 320 over Network 330 to retrieve "toc.smi," a synchronized multimedia 20 integration language ("SMIL") descriptor file defining a set of synchronized plug-ins to run simultaneously in the RealPlayer™ environment. This file contains a reference to "toc.rj," an extensible markup language ("xml") descriptor file for the table of contents extension module compatible with the RealPlayer™.

WO 00/79365

Based on the MIME type of the toc.rj file, RealPlayer™ 301 instantiates SAL 310 and routes the toc.rj file to it. SAL 310 includes File Format Filter 311 and Renderer 312, each implemented using Java™ code linked with native platform C++ code.

5

File Format Filter 311 parses the toc.rj file and hands it over to Renderer 312 through a standard protocol defined by the RealPlayer™. The parsed data contains following elements:

10

(1) name of the Java™ class to instantiate; (2) location from which the Java™ class can be retrieved; and (3) any other parameters specific to the extension module. A sample toc.rj file appears in Figure 4. The first line (starting with *classid*) specifies the Java™ class names; the next two lines (*codebase* and *archive*) specify a URL location and names of Java™ class files; the following line assigns the *width* and the *height* of the window for the table of contents; *sync* is the period in milliseconds between synchronizing events, i.e., between timing calls from the RealPlayer™ to the SAL; *duration* refers to total length of the time-line of the presented event; and, lastly, *param* is the parameter that points to the URL with the toc.xml file, the XML file descriptor of the specific table of contents.

15

Renderer 312 instantiates JVM (Java™ Virtual Machine) 313 needed to run the Java™ code, and also retrieves ".jar" files (packaged Java™ class files) over Network 330. Note that the

URL for the ".jar" files is available from the parsed toc.rj file discussed above. Renderer also instantiates an instance of RealTOC object 314 and hands to it the URL for toc.xml, the XML file describing the specific table of contents object associated 5 with the multimedia presentation. This file includes the nodes and sub-nodes within hierarchical structure of the specific table of contents; it also includes the "begin" and "end" times corresponding to each node and sub-node. RealTOC retrieves the toc.xml file, parses it, and builds the specific table of 10 contents tree structure. A sample toc.xml file appears in Figure 5.

At the beginning of the multimedia presentation, RealVideo object 302 creates a video window for rendering video, while RealTOC creates a TOC window for rendering the table of contents. 15 Both windows are rendered within the larger RealPlayer™ window. As the time-line increases during the presentation, RealPlayer™ 301 periodically calls SAL 310 with the current time; SAL 310 passes the current time to JVM 313, which in turn notifies RealTOC 314; RealTOC 314 highlights the appropriate node on the 20 table of contents tree rendered in the TOC window. A similar process can be constructed with the SAL keeping current time and calling the RealPlayer™ with time updates.

When the user clicks on a particular heading within the table, RealTOC 314 sends the "begin" time associated with the

WO 00/79365

heading to the "seek" function of RealPlayer™ 301 through JVM 313 and SAL 310; RealPlayer™ 301 notifies all synchronized media servers (not shown) of the new "begin" time, waits for the all the media servers to synchronize to the new time, then updates its internal current time and sends the new current time to its extension modules, i.e., RealVideo 302 and SAL 310 (and RealText 103, if applicable); SAL 310 updates the current time of RealTOC 103 through JVM module 313. Importantly, both RealTOC 314 and 314 through JVM module 313. Importantly, both RealTOC 314 and 10 RealVideo 302 are both synchronized to the current time of RealPlayer™ 301. Thus, after RealTOC 314 invokes the seek function requesting a new time, its current time does not skip to the new time until RealPlayer™ 301 notifies RealTOC 314 of the change in current time. If the seek function in RealPlayer™ 301 15 is disabled, as it should be for live video, the time line continues uninterrupted.

Seek function is only one example of controlling the flow of data. Fast forward and rewind functions also control the flow of data by changing the rate of the flow. Moreover, as discussed previously, the data can be arranged in multiple dimensions; 20 then, its flow direction is a vector whose direction may be controlled from the extension module in a way similar to the control of data flow from the table of contents described above. Although a specific embodiment of this invention has been 25 shown and described, it will be understood that various

modifications may be made without departing from the spirit of  
this invention.

**I CLAIM:**

1. A method for extending an extensible framework comprising the steps of:

5 providing the extensible framework;

providing an extension module for the extensible framework;

and

providing an abstraction layer overlaying the extensible framework, the abstraction layer including a uniform cross-platform interface between the extension module and the extensible framework.

10 2. A method for extending an extensible framework according to Claim 1, wherein the abstraction layer is compatible

15 with a second extensible framework and the uniform cross-platform interface is framework-independent.

15 3. A method for extending a first extensible ordered data flow-based framework comprising the steps of:

20 providing the first extensible ordered data flow-based framework;

providing an extension module for the first extensible ordered data flow-based framework;

25 providing an abstraction layer overlaying the first extensible ordered data flow-based framework, the abstraction

layer including a uniform cross-platform interface between the extension module and the first extensible ordered data flow-based framework; and

5 synchronizing the abstraction layer to data flow in the first extensible ordered data flow-based framework.

4. A method for extending a first extensible ordered data flow-based framework according to Claim 3, wherein data of the data flow is arranged in time progression.

10

5. A method for extending a first extensible ordered data flow-based framework according to Claim 4, wherein the first extensible ordered data flow-based framework is a multimedia presentation application, the method further comprising the step  
15 of rendering data-types included in the data.

15

6. A method for extending a first extensible ordered data flow-based framework according to Claim 4, wherein said synchronizing step includes the step of the first extensible  
20 ordered data flow-based framework calling the extension module with current time.

20

7. A method for extending a first extensible ordered data flow-based framework according to Claim 4, wherein said

WO 00/79365

synchronizing step includes the step of the extension module calling the first extensible ordered data flow-based framework with current time.

5       8. A method for extending a first extensible ordered data flow-based framework according to Claim 3, wherein data of the data flow is ordered in multiple dimensions.

10      9. A method for extending a first extensible ordered data flow-based framework according to Claim 3, further including the step of the first extensible ordered data flow-based framework calling the extension module with current position of data.

15      10. A method for extending a first extensible ordered data flow-based framework according to Claim 3, further including the step of the extension module calling the first extensible ordered data flow-based framework with current position of data.

20      11. A method for extending a first extensible ordered data flow-based framework according to Claim 3, further including the step of the first extensible ordered data flow-based framework calling the extension module with current rate of flow of data.

12. A method for extending a first extensible ordered data flow-based framework according to Claim 3, further including the step of the extension module calling the first extensible ordered data flow-based framework with current rate of flow of data.

5

13. A method for extending a first extensible ordered data flow-based framework according to Claim 3, further including the step of the first extensible ordered data flow-based framework calling the extension module with current direction of flow of data.

10

14. A method for extending a first extensible ordered data flow-based framework according to Claim 3, further including the step of the extension module calling the first extensible ordered data flow-based framework with current direction of flow of data.

15

15. A method for extending a first extensible ordered data flow-based framework according to Claim 3, wherein data of the data flow represents discrete states.

20

16. A method for extending a first extensible ordered data flow-based framework according to any one of Claims 3-15, wherein the abstraction layer includes a mechanism for executing platform-independent code.

WO 00/79365

17. A method for extending a first extensible ordered data flow-based framework according to any one of Claims 3, 5, 6, or 7, wherein data of the data flow is packetized and transported to the first extensible ordered data flow-based framework over a network, and the abstraction layer includes a mechanism for executing platform-independent code.

5  
18. A method for extending a first extensible ordered data flow-based framework according to any one of Claims 4, or 8-15, wherein the data of the data flow is packetized and transported to the first extensible ordered data flow-based framework over a network, and the abstraction layer includes a mechanism for executing platform-independent code.

10  
15  
19. A method for extending a first extensible ordered data flow-based framework according to any one of Claims 3-15, wherein the abstraction layer includes a Java™ Virtual Machine.

20  
20. A method for extending a first extensible ordered data flow-based framework according to any one of Claims 3, 5, 6, or 7, wherein data of the data flow is packetized and transported to the first extensible ordered data flow-based framework over a network, and the abstraction layer includes a Java™ Virtual Machine.

21. A method for extending a first extensible ordered data flow-based framework according to any one of Claims 4, or 8-15, wherein the data of the data flow is packetized and transported to the first extensible ordered data flow-based framework from a server storing the data over a network, and the abstraction layer includes a Java™ Virtual Machine.

5  
22. A method for extending a first extensible ordered data flow-based framework according to Claim 21, wherein the first extensible ordered data flow-based framework is a RealPlayer.

10  
15  
23. A method for extending a first extensible ordered data flow-based framework according to any one of Claims 3-15, wherein the abstraction layer is compatible with a second extensible ordered data flow-based framework and the uniform cross-platform interface is framework-independent.

20  
24. A method for extending an extensible ordered data flow-based framework comprising the steps of:  
providing the extensible ordered data flow-based framework;  
providing an extension module for the extensible ordered data flow-based framework;  
providing an abstraction layer overlaying the extensible ordered data flow-based framework, the abstraction layer

including means for uniform cross-platform framework-independent interconnection between the extension module and the extensible ordered data flow-based framework; and

step for synchronizing the abstraction layer to data flow in

5 the extensible ordered data flow-based framework.

25. Apparatus for using an ordered file through applications including an extensible ordered data flow-based framework, an extension module, and an abstraction layer having a 10 uniform cross-platform interface between the extensible ordered data flow-based framework and the extension module, the apparatus comprising:

a computing machine running the extensible ordered data flow-based framework, the extension module, and the abstraction 15 layer;

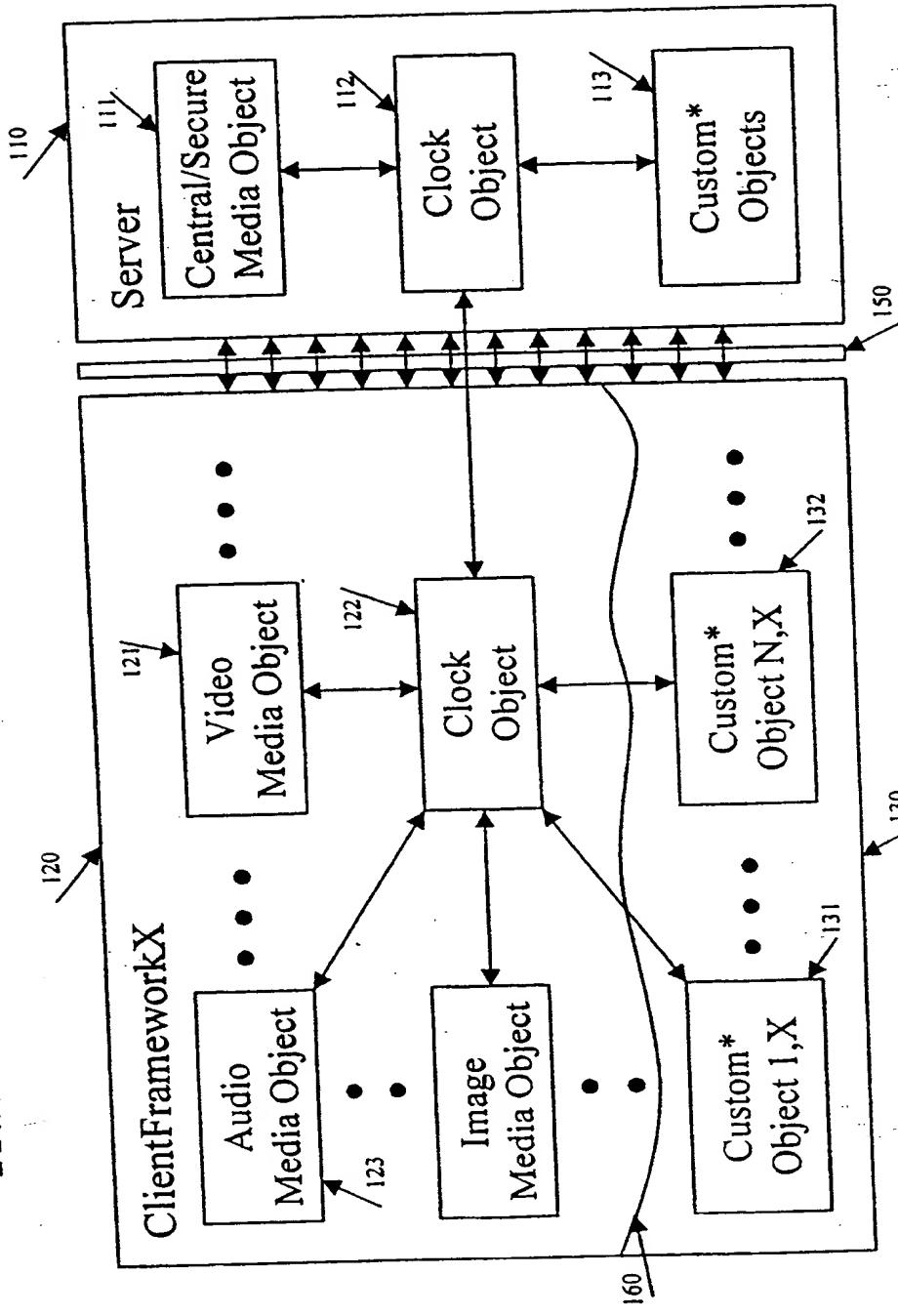
means for synchronizing the extension module and the extensible ordered data flow-based framework;

means for providing content of the ordered file to the extensible ordered data flow-based framework.

20  
26. Apparatus for using an ordered file according to Claim 25, further including means for rendering data-types included in the ordered file.

27. Apparatus for using an ordered file according to Claim 25, wherein the means for providing content includes a network connection and a first server storing the ordered file.

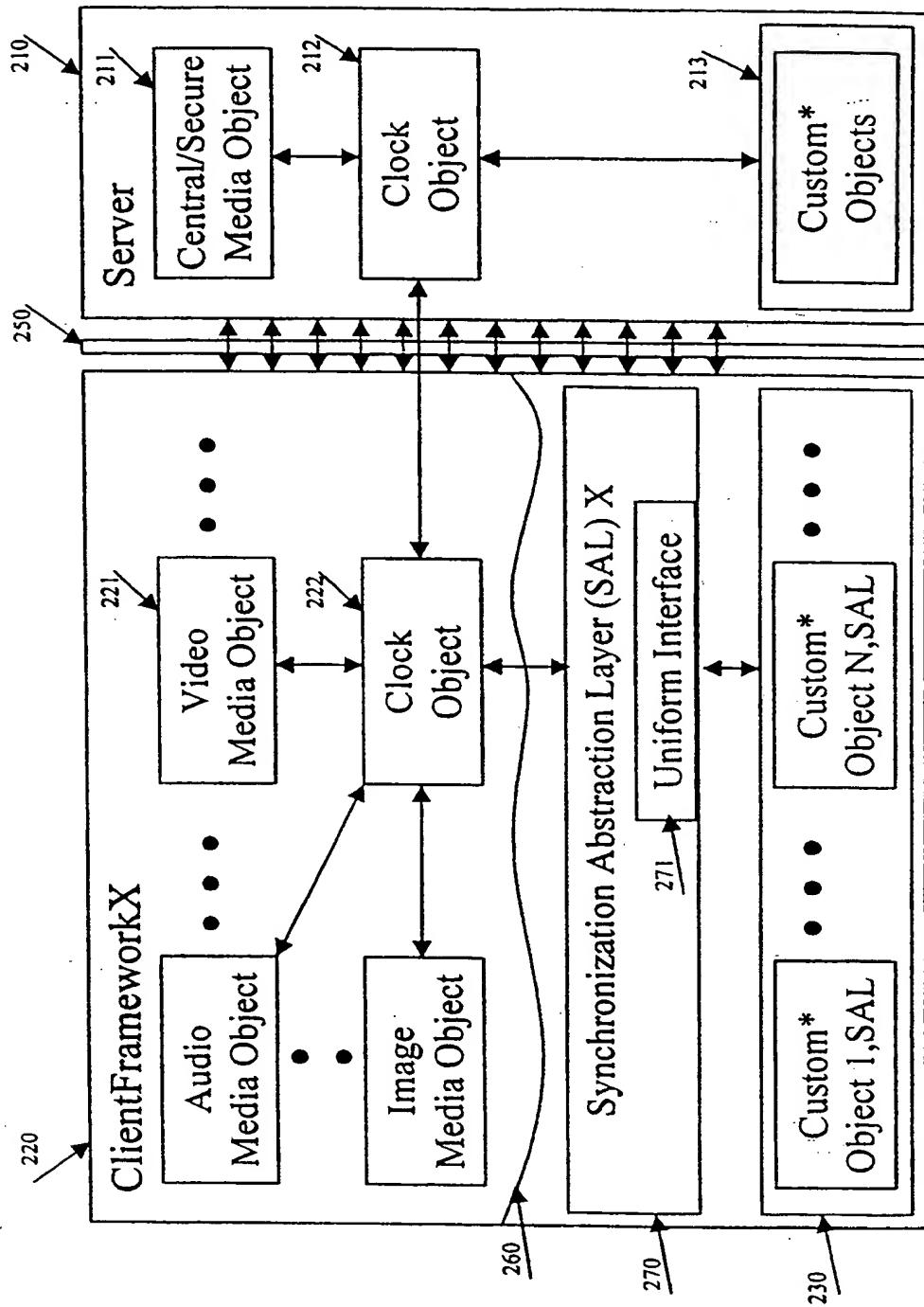
## Traditional Extensible Time-Based Framework



\* Extensible/Programmable/Executable specific to Framework X

Figure 1 - Prior Art

## Uniform Synchronization within Extensible Time-Based Framework



\* Extensible/Programmable/Executable independent of Framework X

Figure 2

RealTOC - An Example Using RealJava Technology

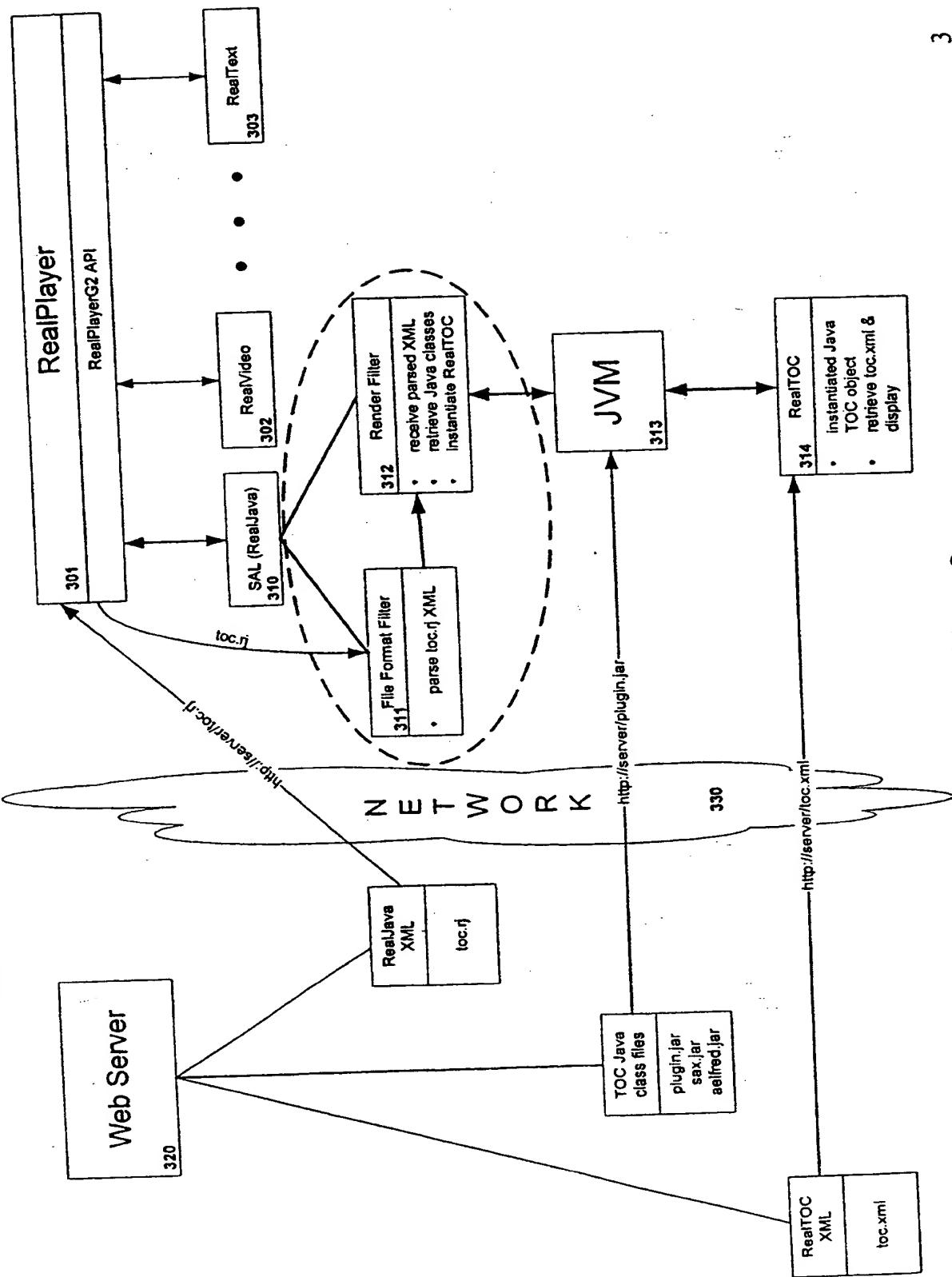


Figure 3

Figure 4  
(toc.rj)

```
<?xml version="1.0"?>
<object
    classid="com.softcom.realjava.plugins.RealTOC"
    codebase="../../website/demos/plugins/codebase/"
    archive="plugin.jar,sax.jar,aelfred.jar"
    width="352" height="350"
    sync="1000"
    duration="900000"
    >
    <param name="URL" value="toc.xml"/>
</object>
```

Figure 5  
(toc.xml)

```
<?xml version="1.0"?>
<TOC>
    <NODE BEGIN="0ms" END="38684ms">
        <TITLE>Acknowledgements</TITLE>
    </NODE>
    <NODE BEGIN="38685ms" END="57627ms">
        <TITLE>Body Temperature</TITLE>
    </NODE>
    <NODE BEGIN="57628ms" END="68999ms">
        <TITLE>Exercise</TITLE>
        <NODE BEGIN="69000ms" END="96757ms">
            <TITLE>Thermoregulatory System</TITLE>
        </NODE>
    </NODE>
    <NODE BEGIN="96758ms" END="126114ms">
        <TITLE>Course Overview</TITLE>
    </NODE>
    <NODE BEGIN="126115ms" END="319999ms">
        <TITLE>Body Temperature</TITLE>
    </NODE>
    <NODE BEGIN="320000ms" END="323335ms">
        <TITLE>Heat Related Illness</TITLE>
        <NODE BEGIN="323336ms" END="371783ms">
            <TITLE>Heat Fatigue</TITLE>
        </NODE>
        <NODE BEGIN="371784ms" END="398749ms">
            <TITLE>Heat Cramps</TITLE>
        </NODE>
        <NODE BEGIN="398750ms" END="437021ms">
            <TITLE>Heat Collapse</TITLE>
        </NODE>
        <NODE BEGIN="437022ms" END="510601ms">
            <TITLE>Heat Exhaustion</TITLE>
        </NODE>
        <NODE BEGIN="510602ms" END="560999ms">
            <TITLE>Heat Stroke</TITLE>
            <NODE BEGIN="561000ms" END="590898ms">
                <TITLE>Symptoms</TITLE>
            </NODE>
            <NODE BEGIN="590899ms" END="654966ms">
                <TITLE>Treatment</TITLE>
            </NODE>
        </NODE>
    </NODE>
    <NODE BEGIN="654967ms" END="900000ms">
        <TITLE>Precautions</TITLE>
    </NODE>
</TOC>
```

Figure 6  
(toc.smil)

```
<smil>
  <head>
    <layout>
      <region id="videoRegion" left="0" top="0"
width="176" height="120"
          fit="meet"/>
      <region id="tocRegion" left="0" top="120"
width="176" height="200"
          fit="fill"/>
    </layout>
  </head>
  <body>
    <par>

      <video id="video"
src="rtsp://demo.softcom.com/heat/video/heatsuresmooth.rm"
region="videoRegion"/>

      <ref id="toc" src="toc.rj" region="tocRegion"
fill="freeze"/>

    </par>
  </body>
</smil>
```

## APPENDIX A

```
5  /* $Header: /cvs/RealJava/src/rjavar/java/com/softcom/realjava/plugins/RealTOC.java,v 1.17
   1999/05/20 20:47:42 aw Exp $ */
   // Copyright (c) 1998 SoftCom, Inc. All Rights Reserved.

 10 package com.softcom.realjava.plugins;

 15 import javax.swing.*;
 16 import javax.swing.tree.*;
 17 import java.awt.*;
 18 import java.awt.event.*;
 19 import java.io.IOException;
 20 import java.net.URL;
 21 import java.net.MalformedURLException;
 22 import org.xml.sax.*;
 23 import com.microstar.xml.SAXDriver;
 24 import com.softcom.realjava.*;
 25 import com.softcom.realjava.time.*;

 26 /**
 27 * Synchronized table of contents plugin.
 28 * Displays a table of contents (TOC) whose nodes are highlighted
 29 * in sync with the presentation. Clicking on a node seeks the
 30 * presentation to that nodes time.
 31 * <P>
 32 * <CODE>RealTOC</CODE> understands a <CODE>URL</CODE> param which
 33 * refers to an XML document specifying a hierarchical TOC.
 34 * <P>
 35 * Sample XML object element:
 36 * <P><TABLE BORDER=1><TR><TD>
 37 * <PRE>
 38 * &lt;?xml version="1.0"?&gt;
 39 * &lt;object
 40 *   classid="com.softcom.realjava.plugins.RealTOC"
 41 *   archive="plugin.jar,sax.jar,aelfred.jar"
 42 *   width="250" height="235"
 43 *   sync="1000"
 44 *   duration="900000"
 45 *   &gt;
 46 *     &lt;param name="URL" value="toc.xml"/&gt;
 47 *   &lt;/object&gt;</PRE>
 48 * </TD></TR></TABLE><P>
 49 * This is the <CODE>TOC</CODE> XML DTD.
```

\* <CODE> BEGIN </CODE> and <CODE> END </CODE> are times specified  
\* in the format documented in <CODE> TimeSpanRegistry.parseTime() </CODE>.  
\* Both are optional, but if either is specified, then both must be specified.  
\* <P> <TABLE BORDER=1> <TR> <TD>  
5 \* <PRE>  
\* &lt;!ELEMENT TOC (NODE+)&gt;  
\* &lt;!ELEMENT NODE (TITLE, NODE\*)&gt;  
\* &lt;!ATTLIST NODE  
\* BEGIN CDATA #IMPLIED  
10 \* END CDATA #IMPLIED  
\* &gt;  
\* &lt;!ELEMENT TITLE (#PCDATA)&gt; </PRE>  
\* </TD> </TR> </TABLE> <P>  
\* Sample <CODE>TOC</CODE> XML document:  
15 \* <P> <TABLE BORDER=1> <TR> <TD>  
\* <PRE>  
\* &lt;?xml version="1.0"?&gt;  
\* &lt;TOC&gt;  
\* &lt;NODE BEGIN = "0" END = "4.999"&gt;  
20 \* &lt;TITLE&gt;Root node one&lt;/TITLE&gt;  
\* &lt;NODE BEGIN = "5" END = "9.999"&gt;  
\* &lt;TITLE&gt;Node one child&lt;/TITLE&gt;  
\* &lt;NODE BEGIN = "10" END = "14.999"&gt;  
\* &lt;TITLE&gt;Node one subchild 1&lt;/TITLE&gt;  
25 \* &lt;/NODE&gt;  
\* &lt;NODE&gt;  
\* &lt;TITLE&gt;Node one subchild 2&lt;/TITLE&gt;  
\* &lt;/NODE&gt;  
\* &lt;/NODE&gt;  
\* &lt;/NODE&gt;  
30 \* &lt;/NODE&gt;  
\* &lt;/NODE&gt;  
\* &lt;/TOC&gt; </PRE>  
\* </TD> </TR> </TABLE> <P>  
\* RealTOC uses the AElfred XML parser and the SAX XML API.  
\* See <A TARGET=\_top  
35 HREF="http://www.microstar.com/aelfred.html" > http://www.microstar.com/aelfred.html </A>  
\* for information on the AElfred XML parser.  
\* See <A TARGET=\_top  
HREF="http://www.microstar.com/sax.html" > http://www.microstar.com/sax.html </A>  
40 \* for information on the SAX API to XML parsers.  
\*/  
public class RealTOC extends JScrollPane implements Plugin {  
  
 private static final String MESSAGE\_CATALOG =  
45 "com.softcom.realjava.plugins.RealTOCMessages";

```
private PluginContext m_pcContext;
private JTree m_jcTree;
private TimeSpanRegistry m_tsRegistry = new TimeSpanRegistry();

5      private static final String DTD = "toc.dtd";

     // Implements Plugin
    public Synchronized getSynchronized() {
        return m_tsRegistry;
    }

10     // Implements Plugin
    public void startPlugin(PluginContext pc) {
        m_pcContext = pc;

15     // Construct AElfred SAX driver
        Parser parser = new SAXDriver();

        // Set parser to handle the XML document.
        parser.setDocumentHandler(new TOCParser());

20     String strURL = m_pcContext.getParameter("URL");
        try {
            if (strURL == null)
                throw new MalformedURLException();
            URL url = new URL(m_pcContext.getDocumentBase(), strURL);
            InputSource is = new InputSource(url.openStream());

            // Pass URL to DTD
            is.setSystemId(getClass().getResource(DTD).toString());

30             // Parse the document
            parser.parse(is);
        } catch (MalformedURLException e) {
            Console.showConsole();
            System.err.println(MessageCatalog.getMessage(MESSAGE_CATALOG,
getClass(), "msg.inf.invalidURL", strURL));
        } catch (IOException e) {
            Console.showConsole();
            System.err.println(MessageCatalog.getMessage(MESSAGE_CATALOG,
getClass(), "msg.inf.parse"));
            e.printStackTrace();
        } catch (SAXException e) {
            Console.showConsole();
            System.err.println(MessageCatalog.getMessage(MESSAGE_CATALOG,
getClass(), "msg.inf.parse"));
        }
    }
}
```

```
Exception ee = e.getException();
if (ee == null)
    e.printStackTrace();
else
    ee.printStackTrace();
5
}

// Add the tree
getViewport().add(m_jcTree, BorderLayout.CENTER);
10

// Implements Plugin
public void destroyPlugin() {
}

15
// Object registered with event registry for each TOC event.
// This object is also set as the TreeNode userObject
private class TOCEvent implements TimeSpanListener {
    // Text to display in tree node
   20
    private String m_strText;

    // -1 if no seek time specified
    private int m_nTime = -1;

    // Path to node this event is associated with
    private TreePath m_tpPath;

    public TOCEvent(DefaultMutableTreeNode tn) {
        m_tpPath = new TreePath(tn.getPath());
    }

    30
    // Set text to display in tree node
    void setText(String strText) {
        m_strText = strText;
    }

    35
    // Set time to seek media to tree node selected
    public void setTime(int nTime) {
        m_nTime = nTime;
    }

    40
    // Return seek time
    public int getTime() {
        return m_nTime;
    }

    45
```

```
// Event time reached, highlight tree node
// Implements TimeSpanListener
public void beginTimeSpan(TimeSpan ts) {
    // Select this tree node. This will make it visible too.
5      m_jcTree.addSelectionPath(m_tpPath);
}

// Implements TimeSpanListener
public void endTimeSpan(TimeSpan ts) {
10     // Deselect this tree node.
     m_jcTree.removeSelectionPath(m_tpPath);
}

// This is used by JTree to draw the node
15     public String toString() {
         return m_strText;
     }
};

20     private class TOCParser extends HandlerBase {
        // Root of tree
        private DefaultMutableTreeNode m_tnRoot;
        // Current parent node
        private DefaultMutableTreeNode m_tnParent;
        // Current child node
25        private DefaultMutableTreeNode m_tnCurrent;

        // Title text accumulator
        private StringBuffer m_sbText = new StringBuffer();
        private boolean m_bAccumulateText = false;

30        // Overrides HandlerBase
        public void startElement(String strName, AttributeList attrs) throws SAXException
        {
            if (strName.equals("TOC")) {
                // Create hidden root of tree
                m_tnRoot = m_tnCurrent = new DefaultMutableTreeNode();
            }
            else if (strName.equals("NODE")) {
40                m_tnParent = m_tnCurrent;

                // Create a new node
                m_tnCurrent = new DefaultMutableTreeNode();

45                // Add node as a child of the current node
                m_tnParent.add(m_tnCurrent);
            }
        }
    }
}
```

```
// Create event for node
TOCEvent te = new TOCEvent(m_tnCurrent);
m_tnCurrent.setUserObject(te);

5 // Set time on event if specified
String strBegin = attrs.getValue("BEGIN");
if (strBegin != null) {
    String strEnd = attrs.getValue("END");
    if (strEnd == null)
        throw new
10 SAXException(MessageCatalog.getMessage(MESSAGE_CATALOG, getClass(),
"msg.ex.missingTime"));

15 try {
    int nBeginTime = TimeSpanRegistry.parseTime(strBegin);
    int nEndTime = TimeSpanRegistry.parseTime(strEnd);
    // Set seek time in event
    te.setTime(nBeginTime);
    // Register event with the TOC timespan registry
    m_tsRegistry.addTimeSpan(new TimeSpan(te, nBeginTime,
20 nEndTime));
    } catch (NumberFormatException e) {
        throw new
25 SAXException(MessageCatalog.getMessage(MESSAGE_CATALOG, getClass(),
"msg.ex.invalidTime"), e);
    }
}
30 else if (strName.equals("TITLE")) {
    // Reset String Buffer for new title
    m_sbText.setLength(0);
    m_bAccumulateText = true;
}
35 // Invalid element
else
    throw new
SAXException(MessageCatalog.getMessage(MESSAGE_CATALOG, getClass(),
40 "msg.ex.invalidElement", strName));
}
// Overrides HandlerBase
public void endElement(String strName) throws SAXException {
    if (strName.equals("TOC")) {
        // Finished building tree. Setup JTree with model.
        // Create the tree with the constructed nodes
45 m_jcTree = new JTree(m_tnRoot);
```

```
m_jcTree.setRootVisible(false);
m_jcTree.setShowsRootHandles(true);

// Allow multiple selection so we can support overlapping timespans
5      m_jcTree.getSelectionModel().setSelectionMode(TreeSelectionModel.DISCONTIGUOUS_TREE_SELECTION);

// When a tree node is clicked (regardless of whether it was selected),
10     // seek the video to the TOCEvent time registered with the node
     m_jcTree.addMouseListener(new MouseAdapter() {
         // XXX mouseClicked is unreliable - it is not always called (bugid
4224704)
         //public void mouseClicked(MouseEvent e) {
         public void mouseReleased(MouseEvent e) {
             // Only handle single clicks
             if (e.getClickCount() != 1)
                 return;
             // Get the path and node clicked on
             20      TreePath path = m_jcTree.getPathForLocation(e.getX(),
e.getY());
             if (path != null) {
                 // Get the event registered with this node
                 TOCEvent te =
5      (TOCEvent)((DefaultMutableTreeNode)path.getLastPathComponent()).getUserObject();
                 if (te != null && te.getTime() >= 0) {
                     // Seek the video
                     m_pcContext.seekPlayer(te.getTime());
                 }
             }
             30      });
         }
     });
     else if (strName.equals("NODE")) {
         // Back up a level
         m_tnCurrent = (DefaultMutableTreeNode)m_tnCurrent.getParent();
     }
     35      else if (strName.equals("TITLE")) {
         // Give title to current node
         ((TOCEvent)m_tnCurrent.getUserObject()).setText(m_sbText.toString());
         m_bAccumulateText = false;
     }
40      }
     //
     // Overrides HandlerBase
     45      public void characters(char ch[], int nStart, int nLength) throws SAXException {
```

```
        if (m_bAccumulateText)
            m_sbText.append(ch, nStart, nLength);
    }
};

5
/*
// XXX debugging
public static void main(String args[]) {
    Frame frm = new Frame();
    frm.setLayout(new BorderLayout());
    RealTOC rt = new RealTOC();
    rt.startPlugin(new PluginContext() {
        public URL getCodeBase() {
            try {
                return new
URL("file:S:/projects/realjava/src/rjavar/plugclasses/");
            } catch (MalformedURLException e) {
                return null;
            }
        }
        public String getParameter(String strParam) {
            return "toc.xml";
        }
        public int getDuration() {
            return 0;
        }
        public void seekPlayer(int nTime) {
        }
    });
    10
    15
    20
    25
    30
    35
    40
    45
    50
    55
    60
    65
    70
    75
    80
    85
    90
    95
    100
    105
    110
    115
    120
    125
    130
    135
    140
    145
    150
    155
    160
    165
    170
    175
    180
    185
    190
    195
    200
    205
    210
    215
    220
    225
    230
    235
    240
    245
    250
    255
    260
    265
    270
    275
    280
    285
    290
    295
    300
    305
    310
    315
    320
    325
    330
    335
    340
    345
    350
    355
    360
    365
    370
    375
    380
    385
    390
    395
    400
    405
    410
    415
    420
    425
    430
    435
    440
    445
    450
    455
    460
    465
    470
    475
    480
    485
    490
    495
    500
    505
    510
    515
    520
    525
    530
    535
    540
    545
    550
    555
    560
    565
    570
    575
    580
    585
    590
    595
    600
    605
    610
    615
    620
    625
    630
    635
    640
    645
    650
    655
    660
    665
    670
    675
    680
    685
    690
    695
    700
    705
    710
    715
    720
    725
    730
    735
    740
    745
    750
    755
    760
    765
    770
    775
    780
    785
    790
    795
    800
    805
    810
    815
    820
    825
    830
    835
    840
    845
    850
    855
    860
    865
    870
    875
    880
    885
    890
    895
    900
    905
    910
    915
    920
    925
    930
    935
    940
    945
    950
    955
    960
    965
    970
    975
    980
    985
    990
    995
    1000
    1005
    1010
    1015
    1020
    1025
    1030
    1035
    1040
    1045
    1050
    1055
    1060
    1065
    1070
    1075
    1080
    1085
    1090
    1095
    1100
    1105
    1110
    1115
    1120
    1125
    1130
    1135
    1140
    1145
    1150
    1155
    1160
    1165
    1170
    1175
    1180
    1185
    1190
    1195
    1200
    1205
    1210
    1215
    1220
    1225
    1230
    1235
    1240
    1245
    1250
    1255
    1260
    1265
    1270
    1275
    1280
    1285
    1290
    1295
    1300
    1305
    1310
    1315
    1320
    1325
    1330
    1335
    1340
    1345
    1350
    1355
    1360
    1365
    1370
    1375
    1380
    1385
    1390
    1395
    1400
    1405
    1410
    1415
    1420
    1425
    1430
    1435
    1440
    1445
    1450
    1455
    1460
    1465
    1470
    1475
    1480
    1485
    1490
    1495
    1500
    1505
    1510
    1515
    1520
    1525
    1530
    1535
    1540
    1545
    1550
    1555
    1560
    1565
    1570
    1575
    1580
    1585
    1590
    1595
    1600
    1605
    1610
    1615
    1620
    1625
    1630
    1635
    1640
    1645
    1650
    1655
    1660
    1665
    1670
    1675
    1680
    1685
    1690
    1695
    1700
    1705
    1710
    1715
    1720
    1725
    1730
    1735
    1740
    1745
    1750
    1755
    1760
    1765
    1770
    1775
    1780
    1785
    1790
    1795
    1800
    1805
    1810
    1815
    1820
    1825
    1830
    1835
    1840
    1845
    1850
    1855
    1860
    1865
    1870
    1875
    1880
    1885
    1890
    1895
    1900
    1905
    1910
    1915
    1920
    1925
    1930
    1935
    1940
    1945
    1950
    1955
    1960
    1965
    1970
    1975
    1980
    1985
    1990
    1995
    2000
    2005
    2010
    2015
    2020
    2025
    2030
    2035
    2040
    2045
    2050
    2055
    2060
    2065
    2070
    2075
    2080
    2085
    2090
    2095
    2100
    2105
    2110
    2115
    2120
    2125
    2130
    2135
    2140
    2145
    2150
    2155
    2160
    2165
    2170
    2175
    2180
    2185
    2190
    2195
    2200
    2205
    2210
    2215
    2220
    2225
    2230
    2235
    2240
    2245
    2250
    2255
    2260
    2265
    2270
    2275
    2280
    2285
    2290
    2295
    2300
    2305
    2310
    2315
    2320
    2325
    2330
    2335
    2340
    2345
    2350
    2355
    2360
    2365
    2370
    2375
    2380
    2385
    2390
    2395
    2400
    2405
    2410
    2415
    2420
    2425
    2430
    2435
    2440
    2445
    2450
    2455
    2460
    2465
    2470
    2475
    2480
    2485
    2490
    2495
    2500
    2505
    2510
    2515
    2520
    2525
    2530
    2535
    2540
    2545
    2550
    2555
    2560
    2565
    2570
    2575
    2580
    2585
    2590
    2595
    2600
    2605
    2610
    2615
    2620
    2625
    2630
    2635
    2640
    2645
    2650
    2655
    2660
    2665
    2670
    2675
    2680
    2685
    2690
    2695
    2700
    2705
    2710
    2715
    2720
    2725
    2730
    2735
    2740
    2745
    2750
    2755
    2760
    2765
    2770
    2775
    2780
    2785
    2790
    2795
    2800
    2805
    2810
    2815
    2820
    2825
    2830
    2835
    2840
    2845
    2850
    2855
    2860
    2865
    2870
    2875
    2880
    2885
    2890
    2895
    2900
    2905
    2910
    2915
    2920
    2925
    2930
    2935
    2940
    2945
    2950
    2955
    2960
    2965
    2970
    2975
    2980
    2985
    2990
    2995
    3000
    3005
    3010
    3015
    3020
    3025
    3030
    3035
    3040
    3045
    3050
    3055
    3060
    3065
    3070
    3075
    3080
    3085
    3090
    3095
    3100
    3105
    3110
    3115
    3120
    3125
    3130
    3135
    3140
    3145
    3150
    3155
    3160
    3165
    3170
    3175
    3180
    3185
    3190
    3195
    3200
    3205
    3210
    3215
    3220
    3225
    3230
    3235
    3240
    3245
    3250
    3255
    3260
    3265
    3270
    3275
    3280
    3285
    3290
    3295
    3300
    3305
    3310
    3315
    3320
    3325
    3330
    3335
    3340
    3345
    3350
    3355
    3360
    3365
    3370
    3375
    3380
    3385
    3390
    3395
    3400
    3405
    3410
    3415
    3420
    3425
    3430
    3435
    3440
    3445
    3450
    3455
    3460
    3465
    3470
    3475
    3480
    3485
    3490
    3495
    3500
    3505
    3510
    3515
    3520
    3525
    3530
    3535
    3540
    3545
    3550
    3555
    3560
    3565
    3570
    3575
    3580
    3585
    3590
    3595
    3600
    3605
    3610
    3615
    3620
    3625
    3630
    3635
    3640
    3645
    3650
    3655
    3660
    3665
    3670
    3675
    3680
    3685
    3690
    3695
    3700
    3705
    3710
    3715
    3720
    3725
    3730
    3735
    3740
    3745
    3750
    3755
    3760
    3765
    3770
    3775
    3780
    3785
    3790
    3795
    3800
    3805
    3810
    3815
    3820
    3825
    3830
    3835
    3840
    3845
    3850
    3855
    3860
    3865
    3870
    3875
    3880
    3885
    3890
    3895
    3900
    3905
    3910
    3915
    3920
    3925
    3930
    3935
    3940
    3945
    3950
    3955
    3960
    3965
    3970
    3975
    3980
    3985
    3990
    3995
    4000
    4005
    4010
    4015
    4020
    4025
    4030
    4035
    4040
    4045
    4050
    4055
    4060
    4065
    4070
    4075
    4080
    4085
    4090
    4095
    4100
    4105
    4110
    4115
    4120
    4125
    4130
    4135
    4140
    4145
    4150
    4155
    4160
    4165
    4170
    4175
    4180
    4185
    4190
    4195
    4200
    4205
    4210
    4215
    4220
    4225
    4230
    4235
    4240
    4245
    4250
    4255
    4260
    4265
    4270
    4275
    4280
    4285
    4290
    4295
    4300
    4305
    4310
    4315
    4320
    4325
    4330
    4335
    4340
    4345
    4350
    4355
    4360
    4365
    4370
    4375
    4380
    4385
    4390
    4395
    4400
    4405
    4410
    4415
    4420
    4425
    4430
    4435
    4440
    4445
    4450
    4455
    4460
    4465
    4470
    4475
    4480
    4485
    4490
    4495
    4500
    4505
    4510
    4515
    4520
    4525
    4530
    4535
    4540
    4545
    4550
    4555
    4560
    4565
    4570
    4575
    4580
    4585
    4590
    4595
    4600
    4605
    4610
    4615
    4620
    4625
    4630
    4635
    4640
    4645
    4650
    4655
    4660
    4665
    4670
    4675
    4680
    4685
    4690
    4695
    4700
    4705
    4710
    4715
    4720
    4725
    4730
    4735
    4740
    4745
    4750
    4755
    4760
    4765
    4770
    4775
    4780
    4785
    4790
    4795
    4800
    4805
    4810
    4815
    4820
    4825
    4830
    4835
    4840
    4845
    4850
    4855
    4860
    4865
    4870
    4875
    4880
    4885
    4890
    4895
    4900
    4905
    4910
    4915
    4920
    4925
    4930
    4935
    4940
    4945
    4950
    4955
    4960
    4965
    4970
    4975
    4980
    4985
    4990
    4995
    5000
    5005
    5010
    5015
    5020
    5025
    5030
    5035
    5040
    5045
    5050
    5055
    5060
    5065
    5070
    5075
    5080
    5085
    5090
    5095
    5100
    5105
    5110
    5115
    5120
    5125
    5130
    5135
    5140
    5145
    5150
    5155
    5160
    5165
    5170
    5175
    5180
    5185
    5190
    5195
    5200
    5205
    5210
    5215
    5220
    5225
    5230
    5235
    5240
    5245
    5250
    5255
    5260
    5265
    5270
    5275
    5280
    5285
    5290
    5295
    5300
    5305
    5310
    5315
    5320
    5325
    5330
    5335
    5340
    5345
    5350
    5355
    5360
    5365
    5370
    5375
    5380
    5385
    5390
    5395
    5400
    5405
    5410
    5415
    5420
    5425
    5430
    5435
    5440
    5445
    5450
    5455
    5460
    5465
    5470
    5475
    5480
    5485
    5490
    5495
    5500
    5505
    5510
    5515
    5520
    5525
    5530
    5535
    5540
    5545
    5550
    5555
    5560
    5565
    5570
    5575
    5580
    5585
    5590
    5595
    5600
    5605
    5610
    5615
    5620
    5625
    5630
    5635
    5640
    5645
    5650
    5655
    5660
    5665
    5670
    5675
    5680
    5685
    5690
    5695
    5700
    5705
    5710
    5715
    5720
    5725
    5730
    5735
    5740
    5745
    5750
    5755
    5760
    5765
    5770
    5775
    5780
    5785
    5790
    5795
    5800
    5805
    5810
    5815
    5820
    5825
    5830
    5835
    5840
    5845
    5850
    5855
    5860
    5865
    5870
    5875
    5880
    5885
    5890
    5895
    5900
    5905
    5910
    5915
    5920
    5925
    5930
    5935
    5940
    5945
    5950
    5955
    5960
    5965
    5970
    5975
    5980
    5985
    5990
    5995
    6000
    6005
    6010
    6015
    6020
    6025
    6030
    6035
    6040
    6045
    6050
    6055
    6060
    6065
    6070
    6075
    6080
    6085
    6090
    6095
    6100
    6105
    6110
    6115
    6120
    6125
    6130
    6135
    6140
    6145
    6150
    6155
    6160
    6165
    6170
    6175
    6180
    6185
    6190
    6195
    6200
    6205
    6210
    6215
    6220
    6225
    6230
    6235
    6240
    6245
    6250
    6255
    6260
    6265
    6270
    6275
    6280
    6285
    6290
    6295
    6300
    6305
    6310
    6315
    6320
    6325
    6330
    6335
    6340
    6345
    6350
    6355
    6360
    6365
    6370
    6375
    6380
    6385
    6390
    6395
    6400
    6405
    6410
    6415
    6420
    6425
    6430
    6435
    6440
    6445
    6450
    6455
    6460
    6465
    6470
    6475
    6480
    6485
    6490
    6495
    6500
    6505
    6510
    6515
    6520
    6525
    6530
    6535
    6540
    6545
    6550
    6555
    6560
    6565
    6570
    6575
    6580
    6585
    6590
    6595
    6600
    6605
    6610
    6615
    6620
    6625
    6630
    6635
    6640
    6645
    6650
    6655
    6660
    6665
    6670
    6675
    6680
    6685
    6690
    6695
    6700
    6705
    6710
    6715
    6720
    6725
    6730
    6735
    6740
    6745
    6750
    6755
    6760
    6765
    6770
    6775
    6780
    6785
    6790
    6795
    6800
    6805
    6810
    6815
    6820
    6825
    6830
    6835
    6840
    6845
    6850
    6855
    6860
    6865
    6870
    6875
    6880
    6885
    6890
    6895
    6900
    6905
    6910
    6915
    6920
    6925
    6930
    6935
    6940
    6945
    6950
    6955
    6960
    6965
    6970
    6975
    6980
    6985
    6990
    6995
    7000
    7005
    7010
    7015
    7020
    7025
    7030
    7035
    7040
    7045
    7050
    7055
    7060
    7065
    7070
    7075
    7080
    7085
    7090
    7095
    7100
    7105
    7110
    7115
    7120
    7125
    7130
    7135
    7140
    7145
    7150
    7155
    7160
    7165
    7170
    7175
    7180
    7185
    7190
    7195
    7200
    7205
    7210
    7215
    7220
    7225
    7230
    7235
    7240
    7245
    7250
    7255
    7260
    7265
    7270
    7275
    7280
    7285
    7290
    7295
    7300
    7305
    7310
    7315
    7320
    7325
    7330
    7335
    7340
    7345
    7350
    7355
    7360
    7365
    7370
    7375
    7380
    7385
    7390
    7395
    7400
    7405
    7410
    7415
    7420
    7425
    7430
    7435
    7440
    7445
    7450
    7455
    7460
    7465
    7470
    7475
    7480
    7485
    7490
    7495
    7500
    7505
    7510
    7515
    7520
    7525
    7530
    7535
    7540
    7545
    7550
    7555
    7560
    7565
    7570
    7575
    7580
    7585
    7590
    7595
    7600
    7605
    7610
    7615
    7620
    7625
    7630
    7635
    7640
    7645
    7650
    7655
    7660
    7665
    7670
    7675
    7680
    7685
    7690
    7695
    7700
    7705
    7710
    7715
    7720
    7725
    7730
    7735
    7740
    7745
    7750
    7755
    7760
    7765
    7770
    7775
    7780
    7785
    7790
    7795
    7800
    7805
    7810
    7815
    7820
    7825
    7830
    7835
    7840
    7845
    7850
    7855
    7860
    7865
    7870
    7875
    7880
    7885
    7890
    7895
    7900
    7905
    7910
    7915
    7920
    7925
    7930
    7935
    7940
    7945
    7950
    7955
    7960
    7965
    7970
    7975
    7980
    7985
    7990
    7995
    8000
    8005
    8010
    8015
    8020
    8025
    8030
    8035
    8040
    8045
    8050
    8055
    8060
    8065
    8070
    8075
    8080
    8085
    8090
    8095
    8100
   
```

THIS PAGE BLANK (USPTO)